Mixed Strand Spaces*

F. Javier THAYER Fábrega Jonathan C. HERZOG
Joshua D. GUTTMAN

The MITRE Corporation
{jt, jherzog, guttman}@mitre.org

Abstract

Strand space analysis *[13, 12] is a method for stating and proving correctness properties for cryptographic protocols. In this paper we apply the same method to the related problem of mixed protocols, and show that a protocol can remain correct even when used in combination with a range of other protocols.*

We illustrate the method with the familiar Otway-Rees [10, 1] protocol. We identify a simple and easily verified characteristic of protocols, and show that the Otway-Rees protocol remains correct even when used in combination with other protocols that have this characteristic.

We also illustrate this method on the Neuman-Stubblebine protocol [9]. This protocol has two parts, an authentication protocol (I) in which a key distribution center creates and distributes a Kerberos-like key, and a re-authentication protocol (II) in which a client resubmits a ticket containing that key. The re-authentication protocol II is known to be flawed [2]. We show that in the presence of protocol II, there are also attacks against protocol I. We then define a variant of protocol II, and prove an authentication property of I that holds even in combination with the modified II.

1 Introduction

In [13, 12, 14], we proposed a general model for encryption protocols and used this model to study specific protocols. In those instances, we assumed that the protocols were being run in a "pure" environment: one in which the protocol is used in isolation. In such an environment, all activity would either be penetrator activity or the activity of a legitimate participant of that protocol.

In practice, however, no environment is "pure." Many different protocols may be in use at the same time, by the same parties, using the same communication channels. As noted in [5], there are at least three reasons that different protocols might use the same secret information:

- Certification is costly, so users will want to use as few certified keys as possible;
- Widespread use of cryptographic APIs will lead to multiple uses of key formats, and perhaps the keys themselves; and
- Smart cards have limited capacities, so cards that are used for multiple protocols might use the same key material for more than one protocol.

Re-use of key material is also a consideration in protocols with multiple parts, such as the Kerberos [6] and the Neuman-Stubblebine [9] protocols. One sub-protocol may be used to retrieve a ticket from a key distribution center, while a second sub-protocol is used to re-present that same ticket to a security-aware server. In such as case, the same secret key is used in two different ways.

In this paper we study the case of mixed protocols, where principals use secret material in more than one protocol. In such cases the two protocols can potentially interact, forming vulerabilities not present in either protocol alone. We apply the strand space model to such cases, and show that the same concepts and techniques as used to analyze the pure environment still apply in that of the mixed.

There have been previous attempts to reason rigorously about protocol interactions. For instance, Meadows [8] studies the Internet Key Exchange protocol, emphasizing the potential interactions among its specific sub-protocols. Gong and Syverson [3] define a (fairly restrictive) class of protocols such that any members of this class may be freely mixed without security failures.

However, our approach is somewhat different. We study a given protocol, which we refer to as the *primary* protocol, and identify some loosely syntactic conditions. We then show that any secondary protocol that meets these syntactic conditions may then freely mix with the primary protocol

*This work was supported by the National Security Agency through US Army CECOM contract DAAB 07-99-C-C201.

Report Documentation Page

Form Approved
OMB No. 0704-0188

1. REPORT DATE	2. REPORT TYPE	3. DATES COVERED
2006		**00-00-2006 to 00-00-2006**

4. TITLE AND SUBTITLE	5a. CONTRACT NUMBER
Mixed Strand Spaces	5b. GRANT NUMBER
	5c. PROGRAM ELEMENT NUMBER
6. AUTHOR(S)	5d. PROJECT NUMBER
	5e. TASK NUMBER
	5f. WORK UNIT NUMBER
7. PERFORMING ORGANIZATION NAME(S) AND ADDRESS(ES) **MITRE Corporation,202 Burlington Road,Bedford,MA,01730-1420**	8. PERFORMING ORGANIZATION REPORT NUMBER
9. SPONSORING/MONITORING AGENCY NAME(S) AND ADDRESS(ES)	10. SPONSOR/MONITOR'S ACRONYM(S)
	11. SPONSOR/MONITOR'S REPORT NUMBER(S)

12. DISTRIBUTION/AVAILABILITY STATEMENT
Approved for public release; distribution unlimited

13. SUPPLEMENTARY NOTES

14. ABSTRACT

15. SUBJECT TERMS

16. SECURITY CLASSIFICATION OF:			17. LIMITATION OF ABSTRACT	18. NUMBER OF PAGES	19a. NAME OF RESPONSIBLE PERSON
a. REPORT	b. ABSTRACT	c. THIS PAGE			
unclassified	**unclassified**	**unclassified**		**11**	

without undermining its secrecy and authentication properties. As we shall see, this sort of result is quite natural given the strand space proof methods. These results fall out from a careful examination of the proofs that the primary protocol meets its security goals in isolation.

In the remainder of this paper, we start (Section 2) by providing a resumé of the strand space theory. We then expand the theory to accomodate the case of mixed protocols (Section 3). We then (Section 4) revisit to a familiar example, the Otway-Rees protocol, which we first studied in [12]. We reproduce some results from [12] in the new context of a mixed environment, and obtain a general contraint which must be met by the other protocols in the environment for Otway-Rees to maintain its correctness properties.

In Section 5, we turn to the Neuman-Stubblebine protocol [9], intended as an example of a protocol with multiple parts. The first part of the Neuman-Stubblebine protocol, called the authentication part, distributes a secret key and a Keberos-like ticket to a client. In the second part of the protocol, called the re-authentication part, the client uses that key and ticket to authenticate itself to a security-aware server.

We perform the same analysis as in Section 4 on the authentication part of the Neuman-Stubblebine protocol, and again obtain a general constraint on other protocols in the environment. We show that the re-authentication does not meet this constraint, and demonstrate one vulnerability that results. We then modify the re-authentication part and show that it meets the general constraint, and so maintains the security of the authentication part.

We end with a brief discussion (Section 6).

2 Strand Spaces

The following is a brief overview of the strand space model as developed in [13], [12], and [14]. Although some theorems and concepts from those two papers are reproduced here, the proofs and proof techniques are not. The reader is referred to those two documents for a more complete and formal exposition. Those already familiar with the strand space method may safely skip this section.

In brief, we introduce a structure called a *strand*, which represents both the abilities of the penetrator and the local experience of a legitimate principal. We then define a structure on strands, called a *bundle*, combines these local views to form a global view. We then define the penetrator, and show that the abilities of the penetrator obey strict bounds. We end with a few words on how these bounds can be used to prove correctness conditions.

More formally:

Definition 2.1 *Let* A *be the set of messages that can be sent between principals. We will call elements of* A terms.

A strand *is a sequence of message transmissions and receptions, where transmission of a term a is represented as $+a$ and reception of term a is represented as $-a$. We will often write a strand as $\langle \pm a_1, \pm a_2, \ldots \pm a_n \rangle$.*

A node *is any is any particular transmission or reception on a particular strand. We often write $\langle s, 1 \rangle$ for the first node on a strand s, $\langle s, 2 \rangle$ for the second, and so on.*

In the case of a legitimate participant, the strand represents those messages that the participant would send or receive as part of one particular run of the protocol. In the case of the penetrator, the strands represent atomic deductions from which more complex actions can be formed. Note that principals are represented only what they say and hear; the penetrator, however, can "say" anything that it can deduce.

Because strands are ordered sequences of message transmissions or receptions, it is meaningful to speak of when something is first said:

Definition 2.2 *Let I be a set of terms. Then a node n is an* entry point *to I if*

1. *the sign of n is positive (i.e. a message transmission),*
2. *the term of n is in I, and*
3. *the term of no previous node on the strand is in I.*

In other words, entry points are those nodes where the strand "enters" the set, i.e. transmits something in the set without having previously transmitted or receiving anything in the set. Entry points are useful for discussing the origins of messages.

We use a similar concept to discuss the first time a particular value is sent out as part of a larger message. To do so, assume that the *subterm* relation is defined on A: $t_1 \sqsubset t_2$ if t_1 is a subterm of t_2.

Definition 2.3 *A term* originates *on a node n iff n is an entry point to the set $I = \{t' : t \sqsubset t'\}$.*

We impose upon strands a graph structure with two types of edges: $\Rightarrow$ and $\rightarrow$. The first arrow represents immediate precedence on a strand:

Definition 2.4 *If n_i and n_{i+1} are consecutive nodes on an strand, we write $n_i \Rightarrow n_{i+1}$.*

The other edge represents inter-strand communication by transmission of terms. When one strands transmits a term, we allow another strand to receive that same term:

Definition 2.5 *If a node $n_1 = +a$ and node $n_2 = -a$, then we write $n_1 \rightarrow n_2$.*

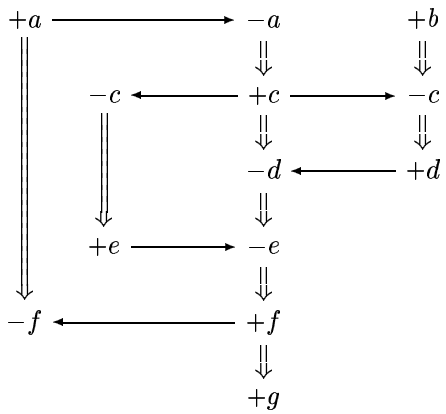


Figure 1. A Bundle

A *strand space*, which is any collection of strands, can be thought of as an ordered graph on the nodes of those strands($\mathcal{N}$) formed by the edges $\rightarrow$ and $\Rightarrow$. A *bundle* is a meaningful finite subgraph of $\langle \mathcal{N}, (\rightarrow \cup \Rightarrow) \rangle$. By "meaningful," we mean that it respects the laws of causality:

- Time flows in only one direction (the arrows of the bundle contain no loops), and
- Causal precedence is preserved. In other words, if an event occurs, then all other events necessary for that occurrence have also occurred. (Bundles are closed backward along arrows).

Definition 2.6 *Let $\mathcal{C} \subset (\rightarrow \cup \Rightarrow)$ be a set of edges, and let $\mathcal{N}_\mathcal{C}$ be the set of nodes incident with any edge in $\mathcal{C}$. $\mathcal{C}$ is a bundle if:*

1. *$\mathcal{C}$ is finite.*
2. *If $n_2 \in \mathcal{N}_\mathcal{C}$ and term(n_2) is negative, then there is a unique $n_1 \in \mathcal{C}$ such that $n_1 \rightarrow n_2$.*
3. *If $n_2 \in \mathcal{N}_\mathcal{C}$ and $n_1 \Rightarrow n_2$ then $n_1 \Rightarrow n_2 \in \mathcal{C}$.*
4. *$\mathcal{C}$ is acyclic.*

(For simplicy, we will often speak of a node being in $\mathcal{C}$ when it is in $\mathcal{N}_\mathcal{C}$.) Figure 1 illustrates an example bundle.

The concept of a bundle is an important one: all possible runs of a protocol can be represented as bundles, and almost all correctness properties can be stated as properties of bundles.

Bundles also have a very useful property:

Lemma 2.7 *Let $\mathcal{C}$ be a set of edges, and let $\preceq_\mathcal{C}$ be the transitive, reflexive closure of $\mathcal{C}$. If $\mathcal{C}$ is a bundle then $\preceq_\mathcal{C}$ is a partial order, i.e. a reflexive, transitive, antisymmetric relation. Then every non-empty set of the nodes of $\mathcal{C}$ have $\preceq_\mathcal{C}$-minimal elements.*

This is also important to our model. Almost all of our reasoning will use the concept of $\preceq_\mathcal{C}$-minimal elements: both entry points and origination points are $\preceq_\mathcal{C}$-minimal elements for sets of certain forms. Before we progress further, however, we add more structure to A and develop our model of the penetrator.

The set of terms A is assumed to be freely generated from two sets:

- $\mathsf{T} \subseteq \mathsf{A}$, which contains texts (the atomic messages), and
- $\mathsf{K} \subseteq \mathsf{A}$, which contains keys (and is disjoint from T),

By two operations:

- $\mathsf{encr} : \mathsf{K} \times \mathsf{A} \rightarrow \mathsf{A}$, which represents encryption, and
- $\mathsf{join} : \mathsf{A} \times \mathsf{A} \rightarrow \mathsf{A}$, which represents concatenation of terms.

We also define injective unary operator $\mathsf{inv} : \mathsf{K} \rightarrow \mathsf{K}$, which maps each member of a public/private key pair to the other, and a symmetric key to itself. We will follow custom and write $\mathsf{inv}(K)$ as K^{-1}, $\mathsf{encr}(K, m)$ as $\{m\}_K$, and $\mathsf{join}(a, b)$ as $a\, b$. If k is a set of keys, k^{-1} denotes the set of inverses of elements of k.

The freeness assumption is stronger than strictly necessary; we assume it here to simply exposition. In [12] we develop the model with weaker freeness assumptions, allowing such relations as associativity of join.

We define the subterm relationship $\sqsubset$ so that for $K \in \mathsf{K}$, $K \sqsubset \{g\}_K$ only if $K \sqsubset g$ already. Defining the subterm relationship in this way reflects an assumption about the penetrator's capabilities: that keys can be obtained from ciphertext only if they are embedded in the text that was encrypted. This might not always be the case—if, for instance, a dictionary attack is possible—but it is the assumption we will make here.

The powers that are available to the penetrator are characterized by two ingredients: a set of keys known initially to the penetrator, and a set of penetrator strands that allow the penetrator to generate new messages.

A *penetrator set* consists of a set of keys $\mathsf{K_P}$ which contains the keys initially known to the penetrator. Typically it would contain: all public keys; all private keys held by the penetrator or his accomplices; and all symmetric keys K_{px}, K_{xp} initially shared between the penetrator and principals playing by the protocol rules. It may also contain "lost keys" that became known to the penetrator previously.

The atomic actions available to the penetrator are encoded in a set of *penetrator strands*. They summarize his ability to discard messages, generate well known messages, piece messages together, and apply cryptographic operations using keys that become available to him. A protocol

attack typically requires several of these atomic actions to be used in combination.

Definition 2.8 *A* penetrator strand *is one of the following:*

M. *Text message:* $\langle +t \rangle$ *where* $t \in \mathsf{T}$

F. *Flushing:* $\langle -g \rangle$

T. *Tee:* $\langle -g, +g, +g \rangle$

C. *Concatenation:* $\langle -g, -h, +g\,h \rangle$

S. *Separation into components:* $\langle -g\,h, +g, +h \rangle$

K. *Key:* $\langle +K \rangle$ *where* $K \in \mathsf{K_P}$.

E. *Encryption:* $\langle -K, -h, +\{h\}_K \rangle$.

D. *Decryption:* $\langle -K^{-1}, -\{h\}_K, +h \rangle$.

Strands that are not penetrator strands are regular strands.

(This set of penetrator strands gives the penetrator powers similar to those in other approaches, e.g. [7, 11].) By explicitly listing the abilities of the penetrator, we gain an important ability ourselves. Because the actions available to the penetrator are independent of any particular protocol, we can prove bounds on the penetrator that are also protocol-independent. In particular, we often show that a set of terms is *honest*:

Definition 2.9 *A set* $I \subseteq A$ *is* honest *relative to a bundle* $\mathcal{C}$ *if and only if whenever a penetrator node* p *is an entry point for* I*,* p *is an* **M** *node or a* **K** *node.*

In other words, a set is honest if elements of that set cannot be synthesized by the penetrator. They can be guessed—by way of a lucky **M** node or **K** node—but the penetrator cannot deduce them via a sequence of decryptions, encryptions, concatenations, or separations.

In applications, honest sets are usually taken to be sets of a particular form, called *ideals*:

Definition 2.10 *If* $\mathsf{k} \subseteq \mathsf{K}$*, a* k*-ideal of* A *is any set* $I \subset \mathsf{A}$ *such that for all* $h \in I$*,* $g \in \mathsf{A}$ *and* $K \in \mathsf{k}$

1. $h\,g, g\,h \in I$.
2. $\{h\}_K \in I$.

The smallest k*-ideal containing* h *is denoted* $I_{\mathsf{k}}[h]$*. If* $S \subseteq \mathsf{A}$*,* $I_{\mathsf{k}}[S]$ *is the smallest* k*-ideal containing* S.

Our main theorem interrelates the structure of ideals with the property of honesty:

Theorem 2.11 *Suppose* $\mathcal{C}$ *is a bundle over* A*;* $S \subseteq \mathsf{T} \cup \mathsf{K}$*;* $\mathsf{k} \subseteq \mathsf{K}$*; and* $\mathsf{K} \subseteq S \cup \mathsf{k}^{-1}$*. Then* $I_{\mathsf{k}}[S]$ *is honest.*

Intuitively, the set S usually contains some number of secrets. The set k usually contains keys which should be considered insecure. Hence, the ideal $I_{\mathsf{k}}[S]$ would represent all terms where a secret occurs in a vulnerable position, i.e. encrypted only with insecure keys. In this case, the theorem states that under certain weak conditions the penetrator will be unable to synthesize elements of the ideal. Hence, if legitimate principals never utter an element of the ideal, then the penetrator is unable to synthesize them:

Corollary 2.12 *Suppose* $\mathcal{C}$ *is a bundle,* $\mathsf{K} = S \cup \mathsf{k}^{-1}$ *and* $S \cap \mathsf{K_P} = \emptyset$*. If no regular node* $n \in \mathcal{C}$ *is an entry point for* $I_{\mathsf{k}}[S]$*, then no node in* $\mathcal{C}$ *is in* $I_{\mathsf{k}}[S]$.

Contrapositively, if the penetrator can deduce an element of the ideal, then some legitimate principal must have slipped and let an element loose:

Corollary 2.13 *Suppose* $\mathcal{C}$ *is a bundle,* $\mathsf{K} = S \cup \mathsf{k}^{-1}$ *and* $S \cap \mathsf{K_P} = \emptyset$*. If there exists a node* $m \in \mathcal{C}$ *such that* m *is in* $I_{\mathsf{k}}[S]$*, then there exists a regular node* $n \in \mathcal{C}$ *such that* n *is an entry point for* $I_{\mathsf{k}}[S]$.

(Note that these are facts about honest sets in general applied to ideals in particular.)

Suppose a key can be proven secret by the above theorem. Then the penetrator is also unable to create any terms that are encrypted with that key:

Theorem 2.14 *Suppose* $\mathcal{C}$ *is a bundle;* $\mathsf{K} = S \cup \mathsf{k}^{-1}$*;* $S \cap \mathsf{K_P} = \emptyset$*; and no regular node* $\in \mathcal{C}$ *is an entry point for* $I_{\mathsf{k}}[S]$*. Then any term of the form* $\{g\}_K$ *for* $K \in S$ *does not originate on a penetrator strand.*

These bounds are usually used in the following way: Suppose that one wishes to prove a correctness condition about a protocol. First, one forms a bundle that reflects the assumptions of the condition in question. Then the penetrator bounds can be used to prove that some other property about the bundle—the conclusion of the correctness condition—must follow.

For example, authentication conditions usually state that if a principal engages in one side of a protocol, then some other principal must have engaged in the other side of the protocol. In our model, local views of a protocol run are represented by regular strands, and global views of a protocol run are represented by bundles. The authentication condition then states that if a bundle contains one particular regular strand, then it must contain another regular strand of a certain form.

Secrecy conditions are more subtle. Because the penetrator is able to say anything that it can deduce, secrecy of a term is shown by proving that it is "unsayable." In particular, it is shown that no regular strand contains entry points to an honest set that contains the secret. Because the set is

honest, no penetrator strand can contain an entry point either. Hence, no strand in the bundle is an entry point to the honest set, and therefore no node is in the set at all. Hence, the set—and in particular, the secret—cannot be said.

3 Multi-Protocol Strand Spaces

We use exactly the same notion of strand space [13, 12, 14]. In the case of mixed protocol environments, however, the regular strands may be those of more than one protocol. We identify one particular protocol for analysis and distinguish the strands of that protocol from all other regular strands:

Definition 3.1 *A mixed strand space is a strand space in which a subset of the regular strands is distinguished. We refer to elements in this set as* primary strands*. The regular strands which are not primary strands are called* secondary strands*. A node is a* primary *or a* secondary node *iff it is on a primary or a secondary strand.*

The intended interpretation of secondary strands is that they correspond to runs of other protocols.

When a strand space mixes protocols, it is typically crucial to correctness to ensure that no secondary strand originates values of some particular form.

Definition 3.2 *A set $I \subseteq \mathsf{A}$ is* unserved *in a strand space Σ if no entry point for I is on a secondary strand in Σ.*

A set $I \subseteq \mathsf{A}$ is strongly unserved *in a strand space Σ if, for every $t \in I$, t does not originate on any secondary strand in Σ.*

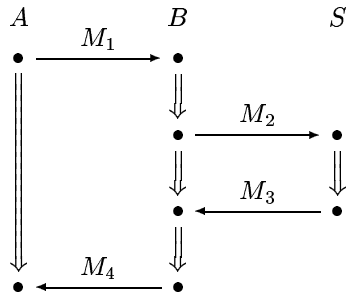
In other words, if a set is unserved, then no "original" instances of a set will occur on secondary strands. They may hear one element of the unserved set and then speak another element, but they may not utter any element of the set without hearing an element first.

The strongly unserved condition is the same in form as the unserved condition, but is strengthened from the set level to the term level. Whereas secondary strands can only speak an element of an unserved set after hearing any other element, they can speak a value in a strongly unserved set—even as a subterm of a larger message—only after receiving that same exact value as a component of some previous message.

4 Mixing Otway-Rees

In this section, we review the Otway-Rees Protocol described and analyzed in [12], of which a normal run is summarized in Figure 2. As in [12], assume the following:

- A set $\mathsf{T}_{\text{name}} \subseteq \mathsf{T}$ of names.



$M_1 = M\, A\, B\, \{N_a\, M\, A\, B\}_{K_{AS}}$

$M_2 = M\, A\, B\, \{N_a\, M\, A\, B\}_{K_{AS}}\, \{N_b\, M\, A\, B\}_{K_{BS}}$

$M_3 = M\, \{N_a\, K_{AB}\}_{K_{AS}}\, \{N_b\, K_{AB}\}_{K_{BS}}$

$M_4 = M\, \{N_a\, K_{AB}\}_{K_{AS}}$

Figure 2. Message Exchange in Otway-Rees

- A mapping $K : \mathsf{T}_{\text{name}} \to \mathsf{K}$. This is intended to denote the mapping which associates to each principal the key it shares with the server. In the literature on this protocol this mapping is usually written using subscripts $K(A) = K_{AS}$.

  We assume the mapping $A \mapsto K_{AS}$ is injective. We also assume $K_{AS} = K_{AS}^{-1}$, i.e. that the protocol is using symmetric cryptography.

  Let L be the set of long-term keys, i.e. the range of K.

We will adopt some conventions on variables for the remainder of this section:

- Variables A, B, C, X range over T_{name};
- Variables K, K' range over K;
- Variables N, M (or the same letters decorated with subscripts) range over $\mathsf{T} \setminus \mathsf{T}_{\text{name}}$, i.e. those texts that are not names.

Other letters such as G and H range over all of A. We would emphasize that N_a is just a variable, having no reliable connection to A, whereas K_{AS} is the result of applying the function K to the argument A. Thus, the latter reliably refers to the long-term key shared between A and S.

4.1 Otway-Rees Formalized

The primary strands for an Otway-Rees strand space may be read off Figure 2. There are only two fine points. First, we assume that the respondent never picks a nonce N_b that happens to be the same as the initiator's nonce N_a. The respondent cannot enforce this directly, because N_a occurs

encrypted with the initiator's long term key; instead, we assume that a probabilistic mechanism enforces it (cf. [12, Section 5]). Second, we assume that the server always selects a session key with three properties: it is a symmetric key; it is unknown to the penetrator; and it is different from any long-term key. The server presumably relies on probabilistic mechanisms to ensure that the last two of these conditions are met.

Definition 4.1 *Let Σ be a strand space.*

1. Init$[A, B, N, M, K]$ *is the set of strands $s \in \Sigma$ whose trace is*

$$\langle + M\ A\ B\ \{N\ M\ A\ B\}_{K_{AS}},\quad -M\ \{N\ K\}_{K_{AS}}\rangle$$

   Σ_{init} *is the union of the range of* Init.

2. Resp$[A, B, N, M, K, H, H']$ *is defined when $N \not\sqsubset H$; its value then is the set of strands in Σ whose trace is*

$$\begin{aligned}\langle &- M\ A\ B\ H,\\ &+ M\ A\ B\ H\ \{N\ M\ A\ B\}_{K_{BS}},\\ &- M\ H'\ \{N\ K\}_{K_{BS}},\\ &+ M\ H'\rangle\end{aligned}$$

   Σ_{resp} *is the union of the range of* Resp.

3. Serv$[A, B, N_a, N_b, M, K]$ *is defined if $K \notin \mathsf{K}_{\mathcal{P}}$, $K \notin \{K_{AS} : A \in \mathsf{T}_{name}\}$ and $K = K^{-1}$; its value then is the set of strands in Σ whose trace is:*

$$\begin{aligned}\langle &- M\ A\ B\ \{N_a\ M\ A\ B\}_{K_{AS}}\ \{N_b\ M\ A\ B\}_{K_{BS}},\\ &+ M\ \{N_a\ K\}_{K_{AS}}\ \{N_b\ K\}_{K_{BS}}\rangle\end{aligned}$$

   Σ_{serv} *is the union of the range of* Serv.

Note that the sets $\Sigma_{\text{serv}}, \Sigma_{\text{init}}, \Sigma_{\text{resp}}$ are pairwise disjoint (cf [12], Lemma 5.2).

For the rest of this example, we will assume that the primary strands are the elements of $\Sigma_{\text{serv}} \cup \Sigma_{\text{init}} \cup \Sigma_{\text{resp}}$ and that the secondary strands are strands of other, unspecified protocols.

Definition 4.2 *Let $\mathsf{L}_0 \subseteq \mathsf{L}$.*

- $\mathsf{Ticket}(\mathsf{L}_0)$ *= the set of all terms of the form $\{N\ K'\}_K$ for $N \in \mathsf{T} \setminus \mathsf{T}_{name}$, $K' \in \mathsf{K}$, and $K \in \mathsf{L}_0$.*
- $\mathsf{Request}(\mathsf{L}_0)$ *= the set of all terms of the form $\{N\ M\ A\ B\}_K$ for $N, M \in \mathsf{T} \setminus \mathsf{T}_{name}$, $A, B \in \mathsf{T}_{name}$, and $K \in \mathsf{L}_0$.*
- $I(\mathsf{L}_0) = I_{\mathsf{k}}[\mathsf{L}_0]$, *where* $\mathsf{k} = (\mathsf{K} \setminus \mathsf{L}_0)$.

An Otway-Rees strand space Σ respects *a set U of principals if, letting $\mathsf{L}_0 = K(U)$ be the image of U under the "key of" mapping K:*

1. $\mathsf{L}_0 \cap \mathsf{K}_{\mathcal{P}} = \emptyset$;
2. $I(\mathsf{L}_0)$ *is unserved in* Σ;
3. $\mathsf{Ticket}(\mathsf{L}_0)$ *and* $\mathsf{Request}(\mathsf{L}_0)$ *are strongly unserved in* Σ.

Otway-Rees remains correct in a mixed protocol environment Σ, for a collection of users U, if Σ respects U. In this paper we will concentrate on a single aspect of the correctness of Otway-Rees, namely the authentication guarantee that Otway-Rees provides to its initiator. However, the secrecy property of Otway-Rees [12, Section 6] and the authentication guarantees it offers to the other participants [12, Section 7.2] may be modified in an equally straightforward way using the same assumptions on Σ.

4.2 Mixed Otway-Rees: Authentication

In this subsection we will prove the authentication guarantee that Otway-Rees provides to its initiator. The proofs are minor modifications of the proofs given in [12].

4.2.1 Preliminaries

We first need a pair of small lemmas. The first is specific to the case of mixed protocols; the second matches a result given in [12].

Proposition 4.3 *Consider a bundle $\mathcal{C}$ in Σ. Suppose $\mathsf{L}_0 \subseteq \mathsf{L}$ is such that $\mathsf{L}_0 \cap K_{\mathcal{P}} = \emptyset$ and $I(\mathsf{L}_0)$ is unserved in Σ. Then no term of the form $\{g\}_K$ for $K \in \mathsf{L}_0$ can originate on a penetrator node in $\mathcal{C}$.*

PROOF. To apply Corollary 2.14, with $S = \mathsf{L}_0$ and $\mathsf{k} = \mathsf{K} \setminus \mathsf{L}_0$, we must check that no regular node n is an entry point for $I_{\mathsf{k}}[S] = I(\mathsf{L}_0)$. By hypothesis, n cannot be a secondary node. n is thus a primary node. However, if n is primary, no long-term key can occur as a subterm of term(n), unless it occurs within the H-term of a responder strand. But in this case n is not an entry point for $I(\mathsf{L}_0)$. ■

We also need a case analysis for the locations at which a term in $\mathsf{Ticket}(\mathsf{L}_0)$ or $\mathsf{Request}(\mathsf{L}_0)$ can originate, assuming that they are originating on a primary strand. The proof matches that of [12, Proposition 7.2 and Corollary 7.3].

Proposition 4.4 *Let s be a primary strand of Σ.*

1. *Suppose $t = \{N\ K\}_{K_{XS}}$ originates on s. Then t and K originate on $\langle s, 2\rangle$, and either $s \in$ Serv$[A, X, N, N', M, K]$ or $s \in$ Serv$[X, B, N', N, M, K]$ for some A, B, N', M.*
2. *Suppose $t = \{N\ M\ A\ B\}_{K_{AS}}$ originates on s, and with $A \neq B$. Then t and N originate on $\langle s, 1\rangle$, and $s \in$ Init$[A, B, N, M, K]$ for some K.*

3. *Suppose* $t = \{N\, M\, A\, B\}_{K_{BS}}$ *originates on* s*, with* $A \neq B$*. Then* t *and* N *originate on* $\langle s, 2\rangle$*, and* $s \in \mathrm{Resp}[A, B, N, M, K, H, H']$*, for some* K*,* H*, and* H'*.*

4.2.2 Initiator's Guarantee

The following theorem asserts that if a bundle contains a strand $s \in \Sigma_{\mathrm{init}}$, then under the expected assumptions, there are primary strands $s_{\mathrm{resp}} \in \Sigma_{\mathrm{resp}}$ and $s_{\mathrm{serv}} \in \Sigma_{\mathrm{serv}}$ which agree on the initiator, responder, and M values.

Theorem 4.5 *Suppose* Σ *respects* U *and* $A, B \in U$*. Suppose* $\mathcal{C}$ *is a bundle in* Σ*;* $A \neq B$*; and* N_a *is uniquely originating in* $\mathcal{C}$*.*

If $s \in \mathrm{Init}[A, B, N_a, M, K]$ *has* $\mathcal{C}$*-height 2, then for some* $N_b \in \mathsf{T}$ *there are primary strands:*

- $s_{serv} \in \mathrm{Serv}[A, B, N_a, N_b, M, K]$ *of* $\mathcal{C}$*-height 2;*
- $s_{resp} \in \mathrm{Resp}[A, B, N_b, M, K, H, H']$ *of* $\mathcal{C}$*-height at least 2, for some* K*,* H*, and* H'*.*

PROOF. The proof of this is similar to the proof of the initiator's guarantee for the unmixed Otway-Rees protocol. The novelty in this case is that we need to establish that a certain term originates on a primary node, whereas in the unmixed case it was sufficient to prove the term originated on a regular node. We will prove this by a sequence of steps. For the remainder of this section, fix Σ, U, $\mathcal{C}$ and s such that the assumptions hold. In particular, by the last assumption of the theorem,

$$\langle +\ M\, A\, B\, \{N_a\, M\, A\, B\}_{K_{AS}},\ -\ M\, \{N_a\, K\}_{K_{AS}}\rangle$$

is the $\mathcal{C}$-trace of a strand s.

Step 1 *There is an* $s_{serv} \in \Sigma_{serv}$ *with* $\mathcal{C}$*-height 2;* s_{serv} *is either of the form* $\mathrm{Serv}[A, X, N_a, N_b, M_1, K]$ *or of the form* $\mathrm{Serv}[X, A, N_b, N_a, M_1, K]$*.*

PROOF. We will apply Proposition 4.3 with $\mathsf{L}_0 = K(U)$, using Definition 4.2, Clauses 1 and 2; it follows $\{N_a\, K\}_{K_{AS}}$ does not originate on a penetrator node in $\mathcal{C}$. Because $\{N_a\, K\}_{K_{AS}} \in \mathsf{Ticket}(\mathsf{L}_0)$ and Σ respects U, by Definition 4.2, Clause 3, it must originate on a primary strand; the node at which it originates is in $\mathcal{C}$. By Proposition 4.4 Clause 1, this node is $\langle s_{\mathrm{serv}}, 2\rangle$ where s_{serv} satisfies one of the conditions:

1. $s_{\mathrm{serv}} \in \mathrm{Serv}[A, X, N_a, N_b, M_1, K]$, or
2. $s_{\mathrm{serv}} \in \mathrm{Serv}[X, A, N_b, N_a, M_1, K]$. ■

Fix $s_{\mathrm{serv}} \in \Sigma_{\mathrm{serv}}$, X, and M_1 satisfying the conditions given in Step 1.

Step 2 $s_{serv} \in \mathrm{Serv}[A, X, N_a, N_b, M_1, K]$.

PROOF. Suppose—in order to derive a contradiction—that $s_{\mathrm{serv}} \in \mathrm{Serv}[X, A, N_b, N_a, M_1, K]$ holds instead. Then $\{N_a\, M_1\, X\, A\}_{K_{AS}}$ is a subterm of $\mathrm{term}(\langle s_{\mathrm{serv}}, 1\rangle)$.

By Proposition 4.3 with $\mathsf{L}_0 = K(U)$ again, using Definition 4.2, Clauses 1 and 2, $\{N_a\, M_1\, A\, X\}_{K_{AS}}$ originates on a regular strand s_1.

Using Clause 3 $\{N_a\, M_1\, X\, A\}_{K_{AS}}$ originates on a primary strand s_1, and by Proposition 4.4, N_a originates on the same strand s_1.

But N_a also originates on the strand we began with, $s \in \mathrm{Init}[A, B, N_a, M, K]$. Because N_a originates uniquely, $s = s_1$. Hence by Proposition 4.4, $X = A = B$, contradicting an assumption. ■

Step 3 $X = B$ *and* $M_1 = M$*.*

PROOF. Since $s_{\mathrm{serv}} \in \mathrm{Serv}[A, X, N_a, N_b, M_1, K]$, $\{N_a\, M_1\, A\, X\}_{K_{AS}} \sqsubset \mathrm{term}(\langle s_{\mathrm{serv}}, 1\rangle)$. By Proposition 4.3 with $\mathsf{L}_0 = K(U)$ again, using Definition 4.2, Clause 1, $\{N_a\, M_1\, A\, X\}_{K_{AS}}$ originates on a regular strand s_1. Using Definition 4.2, Clause 3, s_1 is a primary strand. By Proposition 4.4, N_a originates on the same strand s_1.

But N_a also originates on s. Because N_a originates uniquely, $s = s_1$. Thus $M_1 = M$ and $X = B$, and $s_{\mathrm{serv}} \in \mathrm{Serv}[A, B, N_a, N_b, M, K]$. ■

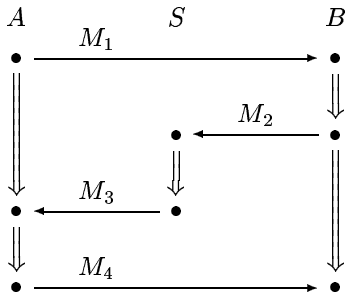
Step 4 *For some* K*,* H*, and* H'*, there is a strand* $s_{resp} \in \mathrm{Resp}[A, B, N_b, M, K, H, H']$ *of* $\mathcal{C}$*-height at least 2.*

PROOF. We again use Proposition 4.3 and Definition 4.2, Clause 3 to infer that $\{N_b\, M\, A\, B\}_{K_{BS}}$ originates on a primary node in $\mathcal{C}$. By Proposition 4.4, this node is the second on a strand $s_{\mathrm{resp}} \in \mathrm{Resp}[A, B, N_b, M, K, H, H']$ for some K, H, and H'. Since $\langle s_{\mathrm{resp}}, 2\rangle \in \mathcal{C}$, it follows s_{resp} has $\mathcal{C}$-height at least 2. ■

5 Neuman-Stubblebine

An important kind of multiple-protocol environment are single protocols that contain multiple parts. Examples of such protocols—such as Kerberos [6], for example—are currently in widespread use. In this paper, we will demonstrate the analysis of such protocols on the Neuman-Stubblebine protocol [9].

The general structural elements needed to describe the protocol are very similar to those of Otway-Rees. In particular, we assume given an injective mapping $K : \mathsf{T}_{\mathrm{name}} \to \mathsf{K}$ which associates to each name a symmetric long term key shared with a central server, and the set L of all long term keys defined to be the range of K. For $K \in \mathsf{L}$, $K^{-1} = K$.



$M_1 = A\, N_a$

$M_2 = B\, \{A\, N_a\, t_b\}_{K_{BS}}\, N_b$

$M_3 = \{B\, N_a\, K\, t_b\}_{K_{AS}}\, \{A\, K\, t_b\}_{K_{BS}}\, N_b$

$M_4 = \{A\, K\, t_b\}_{K_{BS}}\, \{N_b\}_K$

Figure 3. Message Exchange in Neuman-Stubblebine

5.1 The Neuman-Stubblebine Protocol, Part I

The Neuman-Stubblebine protocol starts with an initial authentication protocol, summarized in Figure 3.

In this protocol, K is a unique key generated by the Key Distribution Center S, and t_b is an expiration time for the ticket $\{A\, K\, t_b\}_{K_{BS}}$. (We do not consider the issues of time and timestamps in our analysis.)

First, we define the primary strands to correspond to the three roles of the protocol:

Definition 5.1 *Let Σ be a strand space.*

1. Init$[A, B, N_a, N_b, t_b, K, H]$ *is the set of strands $s \in \Sigma$ of the form:*

$$\langle +\, A\, N_a,\ -\, \{B\, N_a\, K\, t_b\}_{K_{AS}}\, H\, N_b,\ +\, H\{N_b\}_K \rangle$$

   Σ_{init} *is the union of the range of* Init.

2. Resp$[A, B, N_a, N_b, t_b, K]$ *is the set of strands in Σ of the form:*

$$\langle -\, A\, N_a,\ +\, B\, \{A\, N_a\, t_b\}_{K_{BS}}\, N_b,\ -\, \{A\, K\, t_b\}_{K_{BS}}\, \{N_b\}_K \rangle$$

   Σ_{resp} *is the union of the range of* Resp.

3. Serv$[A, B, N_a, N_b, t_b, K]$ *is defined if $K \notin \mathsf{K}_{\mathcal{P}} \cup \mathsf{L}$ and $K = K^{-1}$; its value then is the set of strands in Σ of the form:*

$$\langle -\, B\, \{A\, N_a\, t_b\}_{K_{BS}}\, N_b,\ +\, \{B\, N_a\, K\, t_b\}_{K_{AS}}\, \{A\, K\, t_b\}_{K_{BS}}\, N_b \rangle$$

   Σ_{serv} *is the union of the range of* Serv.

A NS space *is a strand space in which the primary strands are those in Σ_{init}, Σ_{resp}, or Σ_{serv}.*

We will not show that this protocol is correct in all respects. We use the Neuman-Stubblebine protocol as an illustrative example only, and so will instead focus on just one property: the authentication of initiator to responder.

The security of the Neuman-Stubblebine authentication protocol depends upon three types of terms:

1. *Tickets*, which are terms of the form $\{A\, K\, t_b\}_{K_{BS}}$. Tickets are how the secret key is distributed to the responder.

2. *Distributions*, or terms of the form $\{B\, N_a\, K\, t_b\}_{K_{AS}}$. The secret key is distributed to the initiator in terms of this form.

3. *Confirmations*, or terms of the form $\{N_b\}_K$. The initiator finishes the protocol by sending a confirmation to the responder.

We focus on the tickets, distributions, and confirmations built using actual long term keys and session keys; since we do not in general know what values these are, they appear in the definition as the parameters L_0 and S_0. As we use this definition, $\mathsf{k}_0 = \mathsf{L}_0 \cup \mathsf{S}_0$.

Definition 5.2 *Let $\mathsf{k}_0 \subset \mathsf{K}$, $\mathsf{L}_0 \subset \mathsf{L}$ and $\mathsf{S}_0 \subset \mathsf{K} \setminus \mathsf{L}$:*

1. $\mathsf{Ticket}(\mathsf{L}_0)$ *= the set of all terms of the form $\{X\, K\, t_x\}_{K'}$ for $X \in \mathsf{T}_{name}$, $K' \in \mathsf{L}_0$, $K \in \mathsf{K}$, and $t_x \in \mathsf{T} \setminus \mathsf{T}_{name}$.*

2. $\mathsf{Distribute}(\mathsf{L}_0, \mathsf{S}_0)$ *= the set of all terms of the form $\{X\, N\, K\, t_x\}_{K'}$ for $X \in \mathsf{T}_{name}$, $N, t_x \in \mathsf{T} \setminus \mathsf{T}_{name}$, $K' \in \mathsf{L}_0$, and $K \in \mathsf{S}_0$.*

3. $\mathsf{Confirm}(\mathsf{S}_0)$ *= the set of all terms of the form $\{N\}_K$ for $N \in \mathsf{T} \setminus \mathsf{T}_{name}$, and $K \in \mathsf{S}_0$.*

4. $I(\mathsf{k}_0) = I_{\mathsf{k}}[\mathsf{k}_0]$, *where* $\mathsf{k} = (\mathsf{K} \setminus \mathsf{k}_0)$.

5. $\mathsf{SK}(U)$ *= the set of K such that in Σ*

$$\exists A, B \in U\ .\ \text{Serv}[A, B, *, *, *, K] \neq \emptyset$$

Clauses 1 though 3 formalize the terms of interest. Clause 4 is simply a notational convenience. Clause 5, on the other hand, allows us to define a particular set of keys. If we wish to prove a correctness condition about some arbitrary set U of principals, we not only need to consider their long term keys but also the secret keys $\mathsf{SK}(U)$ distributed to any two principals in U, as defined in clause 5.

Definition 5.3 *An NS strand space* Σ respects *a set of principals* U*, if, letting* $\mathsf{L}_0 = K(U)$ *and* $\mathsf{S}_0 = \mathsf{SK}(U)$*:*

1. $(\mathsf{L}_0 \cup \mathsf{S}_0) \cap \mathsf{K}_{\mathcal{P}} = \emptyset$*;*
2. $I(\mathsf{L}_0 \cup \mathsf{S}_0)$ *is unserved in* Σ*;*
3. $\mathsf{Ticket}(\mathsf{L}_0) \cup \mathsf{Distribute}(\mathsf{L}_0, \mathsf{S}_0) \cup \mathsf{Confirm}(\mathsf{S}_0)$ *is strongly unserved in* Σ*.*

Intuitively, a strand space respects a set of principals if it does not interfere with the way the Neuman-Stubblebine protocol uses long term keys, session keys, tickets, distributions and confirmations among members of that set. The long term keys and session keys for those principals must be uncompromised (Clause 1). Secondary strands cannot place any of the above keys in vulnerable positions (Clause 2). Lastly, the tickets, distributions, and confirmations relevant to the principals of interest cannot come from secondary strands (Clause 3). We do not prohibit secondary strands from making terms of those three forms, only from making term of those forms with values that might interfere with those of these principals. For instance, terms of the same forms could safely be constructed using a disjoint set of long term keys.

Before we examine the authentication property of interest, we apply Theorems 2.12 and 2.14 to show two preliminary lemmas: secrecy of keys, and non-synthesis of encrypted terms.

Fix a set of principals U, a NS space Σ, and a bundle $\mathcal{C}$. Let $\mathsf{L}_0 = K(U)$ and $\mathsf{S}_0 = \mathsf{SK}(U)$.

Lemma 5.4 *Suppose* $A, B \in U$*,* Σ *respects* U*, and* K *is uniquely originating. Let* $s_{serv} \in \text{Serv}[A, B, *, *, *, K]$ *be in* $\mathcal{C}$*. For every node* $m \in \mathcal{C}$*,* $m \not\in I(\{K, K_{AS}, K_{BS}\})$*.*

PROOF. Let $\mathsf{k}_0 = \{K, K_{AS}, K_{BS}\}$. By Corollary 2.12 with $S = \mathsf{k}_0$ and $\mathsf{k} = \mathsf{K} \setminus \mathsf{k}_0$, it is sufficient to show that no regular node is an entry point to $I(\mathsf{k}_0)$. Because $I(\mathsf{k}_0)$ is unserved, any regular node which is an entry point to the ideal must be a primary node.

By inspection, no term containing a key originates on any strand in Σ_{init} or Σ_{resp}. However, if $s' \in \Sigma_{\text{serv}}$ then a key originates on node $\langle s', 2\rangle$. So suppose that $\langle s', 2\rangle$ is an entry point to $I(\mathsf{k}_0)$. Then $K \sqsubset \langle s', 2\rangle$, and since K_{AS}, $K_{BS} \not\sqsubset \langle s', 2\rangle$, K originates on s'.

Since K is uniquely originating, and it originates on s_{serv} as well as s', $s' = s_{\text{serv}}$. Moreover, K does not occur in $\langle s, 2\rangle$ unencrypted or encrypted with anything but K_{AS} or K_{BS}. Hence s' does not contain an entry point into $I(\mathsf{k}_0)$, and so no primary strand is an entry point to $I(\mathsf{k}_0)$. ■

Lemma 5.5 *Suppose* Σ *respects* U*. Then no term of the form* $\{g\}_K$ *for* $K \in \mathsf{L}_0$ *can originate on a penetrator node in* $\mathcal{C}$*.*

PROOF. (Similar to that of Proposition 4.3) Apply Corollary 2.14 with $S = \mathsf{L}_0$ and $\mathsf{k} = \mathsf{K} \setminus \mathsf{L}_0$, and confirm that no regular node is an entry point for $I(\mathsf{L}_0)$: Let n be a regular entry point for $I(\mathsf{L}_0)$. Since $I(\mathsf{L}_0) \subset I(\mathsf{L}_0 \cup \mathsf{S}_0)$ is unserved in Σ, n is not a secondary node. By observation, n is not a primary node. ■

We can now prove the authentication condition under consideration:

Theorem 5.6 *Suppose* Σ *respects* U*;* $A, B \in U$*; and* K *is uniquely originating. Suppose* $\mathcal{C}$ *is a bundle in* Σ*, and* $s_1 \in \text{Resp}[A, B, N_a, N_b, t_b, K]$ *has* $\mathcal{C}$*-height 3.*

Then some $s_3 \in \text{Init}[A, B, *, N_b, t_b, K]$ *has* $\mathcal{C}$*-height 3.*

We prove this property by a series of intermediate steps. For those who are uninterested in the details, the statements of each step provide a sketch of the proof.

Step 1 *There is an* $s_2 \in \text{Serv}[A, B, *, *, t_b, K]$ *with* $\mathcal{C}$*-height 2.*

PROOF. $\{A\, K\, t_b\}_{K_{BS}} \sqsubset \langle s_1, 3\rangle$. By Proposition 5.5, $\{A\, K\, t_b\}_{K_{BS}}$ originates on a regular node in $\mathcal{C}$. Because $\{A\, K\, t_b\}_{K_{BS}} \in \mathsf{Ticket}(\mathsf{L}_0)$, that regular node is a primary node n. By inspection, $n = \langle s_2, 2\rangle$ where $s_2 \in \text{Serv}[A, B, *, *, t_b, K]$.

Step 2 *There is an* $s_3 \in \text{Init}[X, Y, N_X, N_b, t_y, K]$ *with* $\mathcal{C}$*-height 3.*

PROOF. By Step 1, $K \in \mathsf{S}_0$. By Proposition 5.4 and Corollary 2.14, no term of the form $\{g\}_K$ originates on a penetrator strand. Hence, $\{N_b\}_K$, which is a subterm of $\langle s_1, 3\rangle$, originates on a regular node $n' \in \mathcal{C}$. Because $\{N_b\}_K \in \mathsf{Confirm}(\mathsf{S}_0)$, n' is a primary node. By inspection, $n' = \langle s_3, 3\rangle$ where $s_3 \in \text{Init}[X, Y, N_X, N_b, t_y, K]$ for some X, Y, N_X, t_y.

Step 3 *There is an* $s_4 \in \text{Serv}[X, Y, *, N_b, t_y, K]$ *with* $\mathcal{C}$*-height 2.*

PROOF. Letting $t = \{Y\, N_b\, K\, t_y\}_{K_{XS}}$, we see that $t \sqsubset \langle s_2, 2\rangle$. If $K_{XS} \not\in \mathsf{L}_0$, then $t \in I(\mathsf{L}_0 \cup \mathsf{S}_0)$, contradicting Proposition 5.4. By Proposition 5.5, t originates on a regular node n''; because $t \in \mathsf{Distribute}(\mathsf{L}_0, \mathsf{S}_0)$, n'' is a primary node.

Inspecting the primary strands, we see that $n'' = \langle s_4, 2\rangle$ where $s_4 \in \text{Serv}[X, Y, *, N_b, t_y, K]$.

Step 4 $s_2 = s_4$

PROOF. K is uniquely originating, and originates on both s_2 and s_4.

Step 5 $s_3 \in \text{Init}[A, B, *, N_b, t_b, K]$ *and* s_3 *has* $\mathcal{C}$*-height 3.*

PROOF. Since $s_2 \in \text{Serv}[A, B, *, N_b, t_b, K]$, $X = A$, $Y = B$, and $t_y = t_b$. In that case, $s_3 = \text{Init}[A, B, *, N_b, t_b, K]$, and it is already established that s_3 has $\mathcal{C}$-height 3.■

In other words, if the responder B finishes a run of the protocol apparently with A, then under the conditions given, A will have finished a run with B.

5.2 Part II (Re-Authentication)

Like Kerberos, this protocol is designed to secure other protocols in which the responder B—which typically provides some networked service—responds to requests from A but keeps no state itself. In such a case, A may need to issue several requests to B and so must re-authenticate itself each time. To that end the Neuman-Stubblebine protocol has a re-authentication part, in which A reuses the ticket issued to it in the initial protocol:

II_1 $A \to B:\ N'_a \{A\, K\, t_b\}_{K_{BS}}$

II_2 $B \to A:\ \{N'_a\}_K\, N'_b$

II_3 $A \to B:\ \{N'_b\}_K$

This re-authentication protocol is known to be flawed on its own [4]. However, it also introduces a potential attack on the initial authentication protocol as well. If B keeps no state—more specifically, if B does not track successful runs of the authentication part of the protocol—then the following attack can be accomplished by starting a run of the re-authentication protocol with B before the initial protocol has finished:

1. $Z(A) \to B:\ A\, N_a$
2. $B \to S:\ B\, \{A\, N_a\, t_b\}_{K_{BS}}\, N_b$
3. $S \to Z(A):\ \{B\, N_a\, K\, t_b\}_{K_{AS}}\, \{A\, K\, t_b\}_{K_{BS}} N_b$

II_1 $\quad Z(A) \to B:\ N_b\, \{A\, K\, t_b\}_{K_{BS}}$

II_2 $\quad B \to Z(A):\ \{N_b\}_K\, N'_b$

4. $Z(A) \to B:\ \{A\, K\, t_b\}_{K_{BS}} \{N_b\}_K$

The attack is possible because a term in $\mathsf{Confirm}(\mathsf{S}_0)$ can now originate on a secondary strand (from the re-authentication part of the protocol). This attack does not seem to be known in the literature. However, it is a pure authentication attack; no session keys (for instance) are divulged.

A variant of the re-authentication part, however, satisfies the conditions of Lemmas 5.4, 5.5, and Theorem 5.6.

II'_1 $A \to B:\ N'_a \{A\, K\, t_b\}_{K_{BS}}$

II'_2 $B \to A:\ \{N'_a\, N'_b\}_K$

II'_3 $A \to B:\ \{A\, N'_b\}_K$

To formalize II', we add a "phantom" starting message in which the initiator receives a copy of message 3 from a run of protocol I. This serves merely to represent the state in which a principal stores the results of a run of I, until ready to begin a run of II'.

Definition 5.7 *Let* Σ *be a strand space.*

1. $\text{ReInit}[A, B, N'_a, N'_b, t_b, K, G, H]$ *is the set of strands in* Σ *of the form:*

$$\begin{aligned} \langle &- \{B\, N_a\, K\, t_b\}_{K_{AS}}\, G\, H, \\ &+ N'_a\, G, \\ &- \{N'_a\, N'_b\}_K, \\ &+ \{A\, N'_b\}_K \rangle \end{aligned}$$

where $N_a \in \mathsf{T}$ *and* $G, H \in \mathsf{A}$. Σ_{reinit} *is the union of the range of* ReInit.

2. $\text{ReResp}[A, B, N'_a, N'_b, t_b, K]$ *is the set of strands in* Σ *of the form:*

$$\begin{aligned} \langle &- \{A\, K\, t_b\}_{K_{BS}}\, N'_a, \\ &+ \{N'_a\, N'_b\}_K, \\ &- \{A\, N'_b\}_K \rangle \end{aligned}$$

Σ_{reresp} *is the union of the range of* ReResp.

A NS+ space *is an infiltrated strand space in which all the regular strands are in* Σ_{init}, Σ_{resp}, Σ_{serv}, Σ_{reinit}, *or* Σ_{reresp}.

Observe that no node on these strands is an entry point to $I(\mathsf{L}_0 \cup \mathsf{k}_0)$. Likewise, $\mathsf{Ticket}(\mathsf{L}_0) \cup \mathsf{Distribute}(\mathsf{L}_0, \mathsf{S}_0) \cup \mathsf{Confirm}(\mathsf{S}_0)$ is strongly unserved by these strands. Hence, we may infer that the modified re-authentication protocol does not interfere with the authentication property given in Theorem 5.6. Setting U to be the set containing A, B, for instance, yields:

Theorem 5.8 *Suppose* $\mathcal{C}$ *is a bundle in a NS+ space, and*

- $s_1 \in \text{Resp}[A, B, N_a, N_b, t_b, K]$ *has* $\mathcal{C}$*-height 3;*
- K_{AS}, K_{BS} *and* $K \notin \mathbf{K}_{\mathcal{P}}$*; and*
- K *is uniquely originating.*

Then $\mathcal{C}$ *contains* $s_3 \in \text{Init}[A, B, *, N_b, t_b, K]$ *with* $\mathcal{C}$*-height 2.*

6 Discussion

Cryptographic protocols are intended to accomplish very specific goals such as authentication or exchange of keys. Analysis of these protocols has usually centered around understanding how well the protocols achieve these stated goals when executed in isolation.

But in fact cryptographic protocols are never executed in isolation. Key exchange is useful only if the keys are then used for some further purpose, such as exchanging data confidentially. Authentication is meaningful only if some particular actions can be performed by the principals, that would not have been permitted had they not been authenticated. These further activities will typically involve the keys or secrets established by the protocol, so there is a risk that these later activities will interfere with the correctness of the base protocol. In many cases, the constraints of practical use mean that an "expensive" protocol is best combined with a "cheaper" protocol, as Kerberos and Neumann-Stubblebine combine one protocol that requires use of a Key Distribution Center with a cheaper re-authentication protocol. Thus, real life is necessarily a case of mixed protocols, even apart from the mixing of independently designed protocols that may be used for unrelated purposes.

In this paper we have developed the simple machinery necessary to reason about this problem within the strand space framework.

References

[1] Michael Burrows, Martín Abadi, and Roger Needham. A logic of authentication. *Proceedings of the Royal Society*, Series A, 426(1871):233–271, December 1989. Also appeared as SRC Research Report 39 and, in a shortened form, in ACM Transactions on Computer Systems 8, 1 (February 1990), 18-36.

[2] John Clark and Jeremy Jacob. A survey of authentication protocol literature: Version 1.0. University of York, Department of Computer Science, November 1997.

[3] Li Gong and Paul Syverson. Fail-stop protocols: An approach to designing secure protocols. In *5th International Working Conference on Dependable Computing for Critical Applications*, pages 44–55, September 1995.

[4] Tzonelih Hwang, Narn-Yoh Lee, Chuang-Ming Li, Ming-Yung Ko, and Yung-Hsiang Chen. Two attacks on Neuman-Stubblebine authentication protocols. *Information Processing Letters*, 53:103–107, 1995.

[5] John Kelsey, Bruce Schneier, and David Wagner. Protocol interactions and the chosen protocol attack. In *Security Protocols, International Workshop April 1997 Proceedings*, pages 91–104. Springer-Verlag, 1998.

[6] J. Kohl and C. Neuman. The Kerberos network authentication service (v5). RFC 1510, September 1993.

[7] Gavin Lowe. Casper: A compiler for the analysis of security protocols. In *10th Computer Security Foundations Workshop Proceedings*, pages 18–30. IEEE Computer Society Press, 1997.

[8] Catherine Meadows. Analysis of the Internet Key Exchange protocol using the NRL protocol analyzer. In *Proceedings, 1999 IEEE Symposium on Security and Privacy*. IEEE Computer Society Press, May 1999.

[9] B. Clifford Neuman and Stuart G. Stubblebine. A note on the use of timestamps as nonces. *Operating Systems Review*, 27(2):10–14, April 1993.

[10] D. Otway and O. Rees. Efficient and timely mutual authentication. *Operating Systems Review*, 21(1):8–10, January 1987.

[11] Lawrence C. Paulson. Proving properties of security protocols by induction. In *10th IEEE Computer Security Foundations Workshop*, pages 70–83. IEEE Computer Society Press, 1997.

[12] F. Javier THAYER Fábrega, Jonathan C. Herzog, and Joshua D. Guttman. Honest ideals on strand spaces. In *Proceedings of the 11th IEEE Computer Security Foundations Workshop*. IEEE Computer Society Press, June 1998.

[13] F. Javier THAYER Fábrega, Jonathan C. Herzog, and Joshua D. Guttman. Strand spaces: Why is a security protocol correct? In *1998 IEEE Symposium on Security and Privacy*. IEEE Computer Society Press, May 1998.

[14] F. Javier THAYER Fábrega, Jonathan C. Herzog, and Joshua D. Guttman. Strand spaces: Proving security protocols correct. *Journal of Computer Security*, 1999. Forthcoming.